On the complexity of strong Nash equilibrium:
Hard–to–solve instances and smoothed complexity

Nicola Gatti and **Marco Rocco**
Politecnico di Milano
Piazza Leonardo da Vinci 32
Milano, Italy
{ngatti, mrocco}@elet.polimi.it

Tuomas Sandholm
Carnegie Mellon
Computer Science Department
5000 Forbes Avenue
Pittsburgh, PA 15213, USA
sandholm@cs.cmu.edu

Abstract

The computational characterization of game–theoretic solution concepts is a central topic in artificial intelligence, with the aim of developing computationally efficient tools for finding optimal ways to behave in strategic interactions. The central solution concept in game theory is *Nash equilibrium* (NE). However, it fails to capture the possibility that agents can form coalitions (even in the 2–agent case). *Strong Nash equilibrium* (SNE) refines NE to this setting. It is known that finding an SNE is $\mathcal{NP}$–complete when the number of agents is constant. This hardness is solely due to the existence of mixed–strategy SNEs, given that the problem of enumerating all pure–strategy SNEs is trivially in $\mathcal{P}$. Our central result is that, in order for a game to have at least one non–pure–strategy SNE, the agents' payoffs restricted to the agents' supports must, in the case of 2 agents, lie on the same line, and, in the case of n agents, lie on an $(n-1)$–dimensional hyperplane. Leveraging this result, we provide two contributions. First, we develop worst–case instances for support–enumeration algorithms. These instances have only one SNE and the support size can be chosen to be of any size—in particular, arbitrarily large. Second, we prove that, unlike NE, finding an SNE is in smoothed polynomial time: generic game instances (i.e., all instances except knife–edge cases) have only pure–strategy SNEs.

1 Introduction

The computational characterization of game–theoretic solution concepts is a central topic in artificial intelligence, with the aim of developing computationally efficient tools for finding optimal ways to behave in strategic interactions. The central solution concept provided by game theory is *Nash equilibrium* (NE). Every finite game admits at least one NE in mixed strategies. Computer scientists have characterized the complexity of NE finding and provided a number of algorithms. Finding an NE of a strategic–form (aka normal-form) game is $\mathcal{PPAD}$–complete [Daskalakis *et al.*, 2006] even with just two agents [Chen *et al.*, 2009]. Such games are called bimatrix games. Although $\mathcal{PPAD} \subseteq \mathcal{NP}$, it is gen-

erally believed that $\mathcal{PPAD} \neq \mathcal{P}$ and therefore that there not exists any polynomial–time algorithm to find an NE unless $\mathcal{P} = \mathcal{NP}$. Furthermore, bimatrix games do not have a fully polynomial–time approximation scheme unless $\mathcal{PPAD} \subseteq \mathcal{P}$ [Chen *et al.*, 2009] and finding an NE in bimatrix games is not in smoothed–$\mathcal{P}$ unless $\mathcal{PPAD} \subseteq \mathcal{RP}$ [Chen *et al.*, 2006] and, therefore, by definition of smoothed complexity, game instances remain hard even if subjected to small perturbations. Instances [Savani and von Stengel, 2006] that require exponential time when solved with a number of algorithms [Lemke and Howson, 1964; Porter *et al.*, 2009] are known. (However, these instances are unstable: they become easy when small perturbations are applied [Gatti *et al.*, 2012].)

NE captures the situation in which no agent can gain more by unilaterally changing her strategy. When agents can form coalitions and change their strategies multilaterally in a coordinated way, the most natural solution concept is *strong Nash equilibrium* (SNE) [Aumann, 1960]. An SNE is a strategy profile from which no coalition can deviate in a way that benefits each of the deviators. SNE has significantly different properties than NE. An SNE is not assured to exist. Finding an SNE (determining if one exists) is $\mathcal{NP}$–complete when the number of agents is constant; $\mathcal{NP}$–hardness was proven in [Conitzer and Sandholm, 2008] and membership in $\mathcal{NP}$ in [Gatti *et al.*, 2013]. Unlike for NE, the literature has very few algorithms for SNE. There are algorithms for finding pure–strategy SNEs for specific classes of games, e.g., congestion games [Holzman and Law-Yone, 1997; Hayrapetyan *et al.*, 2006; Rozenfeld and Tennenholtz, 2006; Hoefer and Skopalik, 2010], connection games [Epstein *et al.*, 2007], maxcut games [Gourvès and Monnot, 2009], and continuous games [Nessah and Tian, 2012]. However, the hardness in the general setting is due to the existence of mixed–strategy SNEs, given that the pure–strategy ones can be found in polynomial time by combining support enumeration and verification. The only prior SNE–finding algorithm that works also for mixed strategies is only for 2–agent games [Gatti *et al.*, 2013]. Its application to instances from the ubiquitous NE benchmark testbed, GAMUT [Nudelman *et al.*, 2004], shows that these instances either admit pure SNEs or do not admit any SNE and therefore that new benchmark testbeds for SNE–finding

algorithms are needed.

In this paper, we provide the following main contributions.

- We show that, if there is a mixed–strategy SNE, then the payoffs restricted to the actions in the support must satisfy specific conditions. For example, in 2–agent games, they must lie on the same line in agents' utilities space.

- We show how to generate 2–agent games with m actions per agent, only one SNE, and any desired number of actions $\{1, \ldots, \frac{m}{2}\}$ in the support of each agent's mixed strategy. These are the worst–case instances for support–enumeration algorithms, requiring time $O(4^{\frac{m}{2}})$.

- We show that, for any number of agents, finding an SNE is in smoothed–$\mathcal{P}$, thus admitting a deterministic support–enumeration algorithm with smoothed polynomial running time. In the generic case (i.e., in all except knife–edge cases), all SNEs are pure.

2 Preliminaries

A strategic–form game is a tuple (N, A, U) where [Shoham and Leyton-Brown, 2008]:

- $N = \{1, \ldots, n\}$ is the set of agents (we denote by i a generic agent),
- $A = \{A_1, \ldots, A_n\}$ is the set of agents' actions and A_i is the set of agent i's actions (we denote a generic action by a, and by m_i the number of actions in A_i),
- $U = \{U_1 \ldots, U_n\}$ is the set of agents' utility arrays where $U_i(a_1, \ldots, a_n)$ is agent i's utility when the agents play actions $a_1, \ldots, a_n$.

We denote by $x_i(a_i)$ the probability with which agent i plays action $a_i \in A_i$ and by $\mathbf{x}_i$ the vector of probabilities $x_i(a_i)$ of agent i. We denote by Δ_i the space of well–defined probability vectors over A_i. We denote by S_i the support of agent i, that is, the set of actions played with positive probability, and by S the support profile $(S_1, \ldots, S_n)$.

The most central solution concept in game theory is NE. A strategy profile $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is an NE if, for each $i \in N$, $\mathbf{x}_i^T U_i \prod_{j \neq i} \mathbf{x}_{-j} \geq \mathbf{x}_i'^T U_i \prod_{j \neq i} \mathbf{x}_{-j}$ for every $\mathbf{x}_i' \in \Delta_i$. Every finite game admits at least one NE in mixed strategies. The problem of finding an NE can be expressed as:

$$v_i - \sum_{a_{-i} \in A_{-i}} U_i(a_i, a_{-i}) \cdot \prod_{\substack{j \in N: \\ j \neq i}} x_j(a_j) \geq 0 \quad \forall i \in N, a_i \in A_i \quad (1)$$

$$x_i(a_i) \cdot \left(v_i - \sum_{a_{-i} \in A_{-i}} U_i(a_i, a_{-i}) \cdot \right.$$
$$\left. \cdot \prod_{\substack{j \in N: \\ j \neq i}} x_j(a_j) \right) = 0 \quad \forall i \in N, a_i \in A_i \quad (2)$$

$$x_i(a_i) \geq 0 \quad \forall i \in N, a_i \in A_i \quad (3)$$

$$\sum_{a_i \in A_i} x_i(a_i) = 1 \quad \forall i \in N \quad (4)$$

Here v_i is the expected utility of agent i. Constraints (1) force the expected utility v_i to be non–smaller than the expected utility given by every action a_i available to agent i. Constraints (2) force the expected utility v_i of agent i to be equal to the expected utility given by every action a_i that is played with positive probability by agent i. Constraints (3) force each probability $x_i(a_i)$ to be nonnegative. Constraints (4) force each probability vector $\mathbf{x}_i$ to sum to one.

An SNE [Aumann, 1960] strengthens the NE concept by requiring the strategy profile to be resilient also to multilateral deviations by any coalition of agents. That is, in an SNE no coalition of agents can deviate in a way that strictly increases the expected utility of each member of the coalition, again keeping the strategies of the agents outside the coalition fixed. So, an SNE combines two notions: an SNE is an NE and it is weakly Pareto efficient over the space of all the strategy profiles for each possible coalition. Unlike an NE, an SNE is not assured to exist even in mixed strategies.

Multi–objective programming provides Pareto optimality conditions [Miettinen, 1999], based on Karush–Kuhn–Tucker (KKT) results. These conditions are:

$$\sum_i \lambda_i \cdot \nabla f_i(\mathbf{z}) + \sum_j \mu_j \cdot \nabla g_j(\mathbf{z}) + \sum_k \nu_k \cdot \nabla h_k(\mathbf{z}) = \mathbf{0} \quad (5)$$

$$\mu_j \cdot g_j(\mathbf{z}) = 0 \quad \forall j \quad (6)$$

$$\lambda_i, \mu_j \geq 0 \quad \forall i, j \quad (7)$$

$$\sum_i \lambda_i = 1 \quad (8)$$

where $f_i(\mathbf{z})$ are the objective functions to minimize, $g_j(\mathbf{z})$ are inequality constraints as $g_j(\mathbf{z}) \leq 0$, and h_k are equality constraints as $h_k(\mathbf{z}) = 0$. The λ_i, μ_j, ν_k are called KKT multipliers; λ_i is the weight of objective function f_i, μ_j is the weight of constraint g_j, and h_k is the weight of constraint h_k.

The KKT conditions (5)–(8) are necessary conditions for local Pareto efficiency [Miettinen, 1999]. For games, we can map these conditions to the case of Pareto efficiency for a single coalition $C \subseteq N$ as follows:

- f_i is agent i's expected utility multiplied by '-1' (given that in KKT the objective f_i is to minimize);
- g_j is a constraint of the form $-x_i(a_i) \leq 0$;
- h_k is a constraint of the form $\sum_{a_i \in A_i} x_i(a_i) - 1 = 0$.

3 Characterization of mixed–strategy SNEs

We study the properties of mixed–strategy SNEs. We focus on the basic case of 2–agent games and we discuss how the reasoning can be extended to games with 3 or more agents.

We denote by P_{mix} and by P_{cor} the sets of points in the agents' utility spaces $\mathbb{E}[U_1] \times \mathbb{E}[U_2]$ that are on the Pareto frontier when the agents play *mixed* and *correlated* strategies, respectively. Obviously, points in P_{cor} non–strictly Pareto dominate points in P_{mix}, given that mixed strategies constitute a subset of correlated strategies. In addition, we denote by $P_{mix}(S)$ and $P_{cor}(S)$ the Pareto frontiers in mixed and correlated strategies, respectively, when the game is restricted to the sets of actions in support profile S.

Theorem 3.1 *Consider a non–degenerate 2–agent game with two actions per agent. If there is a mixed–strategy SNE, then $P_{mix} = P_{cor}$.*

Proof. We can write down the game as follows:

		a₃	a₄
agent 1	a₁	p_1, q_1	p_2, q_2
	a₂	p_3, q_3	p_4, q_4

agent 2

By assumption:
- There is a mixed–strategy NE. Therefore:

$$x_2(a_3) \cdot p_1 + x_2(a_4) \cdot p_2 = x_2(a_3) \cdot p_3 + x_2(a_4) \cdot p_4 \quad (9)$$

$$x_1(a_1) \cdot q_1 + x_1(a_2) \cdot q_3 = x_1(a_1) \cdot q_2 + x_1(a_3) \cdot q_4 \quad (10)$$

- The NE is on P_{mix}, being an SNE. Therefore, the KKT conditions are satisfied (since they are necessary conditions for local weak Pareto efficiency):

$$-\lambda_1 \cdot (x_2(\mathsf{a}_3) \cdot p_1 + x_2(\mathsf{a}_4) \cdot p_2)-$$
$$-\lambda_2 \cdot (x_2(\mathsf{a}_3) \cdot q_1 + x_2(\mathsf{a}_4) \cdot q_2) = \nu_1 \quad (11)$$

$$-\lambda_1 \cdot (x_2(\mathsf{a}_3) \cdot p_3 + x_2(\mathsf{a}_4) \cdot p_4)-$$
$$-\lambda_2 \cdot (x_2(\mathsf{a}_3) \cdot q_3 + x_2(\mathsf{a}_4) \cdot q_4) = \nu_1 \quad (12)$$

$$-\lambda_1 \cdot (x_1(\mathsf{a}_1) \cdot p_1 + x_1(\mathsf{a}_3) \cdot p_3)-$$
$$-\lambda_2 \cdot (x_1(\mathsf{a}_1) \cdot q_1 + x_1(\mathsf{a}_3) \cdot q_3) = \nu_2 \quad (13)$$

$$-\lambda_1 \cdot (x_1(\mathsf{a}_1) \cdot p_2 + x_1(\mathsf{a}_2) \cdot p_4)-$$
$$-\lambda_2 \cdot (x_1(\mathsf{a}_1) \cdot q_2 + x_1(\mathsf{a}_3) \cdot q_4) = \nu_2 \quad (14)$$

By combining (9), (10) with (11)–(14), we obtain

$$x_2(\mathsf{a}_3) \cdot q_1 + x_2(\mathsf{a}_4) \cdot q_2 = x_2(\mathsf{a}_3) \cdot q_3 + x_2(\mathsf{a}_4) \cdot q_4 \quad (15)$$
$$x_1(\mathsf{a}_1) \cdot p_1 + x_1(\mathsf{a}_2) \cdot p_3 = x_1(\mathsf{a}_1) \cdot p_2 + x_1(\mathsf{a}_2) \cdot p_4 \quad (16)$$

By trivial mathematics, we can rewrite (9), (10), (15), (16) as

$$x_2(\mathsf{a}_3) \cdot (p_1 - p_3) = x_2(\mathsf{a}_4) \cdot (p_4 - p_2)$$
$$x_1(\mathsf{a}_1) \cdot (q_1 - q_2) = x_1(\mathsf{a}_2) \cdot (q_4 - q_3)$$
$$x_2(\mathsf{a}_3) \cdot (q_1 - q_3) = x_2(\mathsf{a}_4) \cdot (q_4 - q_2)$$
$$x_1(\mathsf{a}_1) \cdot (p_1 - p_2) = x_1(\mathsf{a}_2) \cdot (p_4 - p_3)$$

We can safely assume $p_1 \neq p_3, p_2 \neq p_4, p_1 \neq p_2, p_3 \neq p_4$, and the analogous inequalities for agent 2, since this assumption excludes only degenerate games. Indeed, if $p_1 = p_3$, then, by the above conditions, we have $p_2 = p_4$, and therefore actions a_1 and a_2 are the same; if $p_1 = p_2$, then, by the above conditions, we have $p_3 = p_4$, and therefore, in order to have a mixed–strategy NE, we need $p_1 = p_2 = p_3 = p_4$. The same reasoning holds for agent 2. Thus, we derive the following conditions:

$$\frac{p_1 - p_2}{p_4 - p_3} = \frac{q_1 - q_2}{q_4 - q_3} \qquad \frac{p_1 - p_3}{p_4 - p_2} = \frac{q_1 - q_3}{q_4 - q_2}$$

We can give a simple geometric interpretation of the above conditions. Call $R_i = (p_i, q_i)$. Each R_i is a point in the space $\mathbb{E}[U_1] \times \mathbb{E}[U_2]$. The above conditions state that:

- $\overline{R_1 R_2}$ is parallel to $\overline{R_3 R_4}$,
- $\overline{R_1 R_3}$ is parallel to $\overline{R_2 R_4}$,

and therefore R_1, R_2, R_3, R_4 are the vertices of a parallelogram, see Fig. 1(a). Given that

- a mixed–strategy NE is strictly inside the parallelogram (it being the convex (non–degenerate) combination of the vertices), see Fig. 1(a), and that
- it must be on a Pareto efficient edge (since, if it is strictly inside the parallelogram—as in Fig. 1(a)—then it is Pareto dominated by some point on some edge),

we have that R_1, R_2, R_3, R_4 must be aligned according to a line of the form $\mathbb{E}[U_1] + \phi \cdot \mathbb{E}[U_2] = const$ with $\phi \in (-1, 0)$, see, e.g., Fig. 1(b). Thus, the combination of R_1, R_2, R_3, R_4 through every mixed–strategy profile lies on the line connecting the two extreme vertices; for example, in Fig. 1(b) the extreme vertices are R_2 and R_1. It trivially follows that $P_{mix} = P_{cor}$. $\qquad \square$

The proof of the above theorem provides necessary conditions for a game to admit a mixed–strategy SNE. These conditions are not sufficient. Indeed, we can show that only some alignments of R_1, R_2, R_3, R_4 can lead to an SNE:
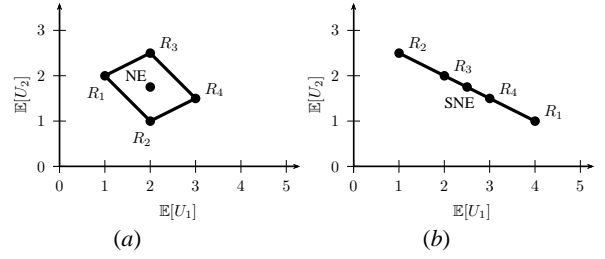

Figure 1: Examples used in the proof of Theorem 3.1.

Corollary 3.2 *The only alignments of R_1, R_2, R_3, R_4 that can lead to an SNE satisfy the following conditions:*
- R_1 *or* R_4 *is one extreme,*
- *moving from the previous extreme, the next vertex is* R_4 *or* R_1,
- *the next vertex is* R_2 *or* R_3,
- R_3 *or* R_2 *is the other extreme.*

Proof. We initially show that it is not possible to have a mixed–strategy NE for alignments different from those considered in the corollary. For reasons of space, we study a single case, the proof in the other cases is similar. Consider, for contradiction, the case in which R_1 is the extreme with the maximum $\mathbb{E}[U_1]$, then the sequence is R_2, R_3, and R_4. It can be observed that the first action of the first agent dominates the second action because $U_1(R_1) > U_1(R_3)$ and $U_1(R_2) > U_1(R_4)$ and therefore the first agent will play the first action with probability one. So, there is no mixed–strategy NE. Instead, for the alignments considered in the corollary, dominance does not apply, as shown, e.g., in Fig. 2. $\qquad \square$
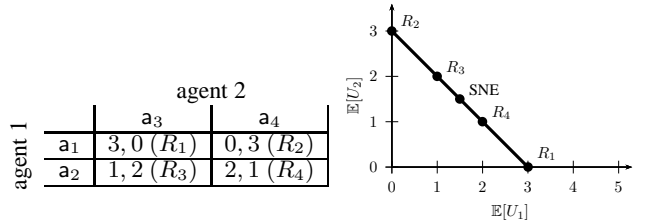

Figure 2: Example 2–agent game (left) and its Pareto frontier (right).

We now extend the previous result to the setting in which each agent has m actions and $|S_1| = |S_2| = 2$.

Corollary 3.3 *Consider a non–degenerate 2–agent game with m actions per agent. If there is a mixed–strategy SNE with support sizes $|S_1| = |S_2| = 2$, then $P_{mix}(S_1, S_2) = P_{cor}(S_1, S_2)$.*

Proof. We can split the NE constraints and KKT conditions into two groups: those generated considering deviations towards pure or mixed strategies over the supports S_1 and S_2 and those generated considering deviations towards pure or mixed strategies over actions off the supports S_1 and S_2. The constraints belonging to the first group are the same as in the case with two actions per agent considered in the proof of Theorem 3.1. The second group overconstrains the problem and it is not necessary for the proof. Thus, restricting the game to the actions in S_1 and S_2, Theorem 3.1 holds and therefore $P_{mix}(S_1, S_2) = P_{cor}(S_1, S_2)$, see, e.g., Fig. 3. $\quad \square$

agent 2

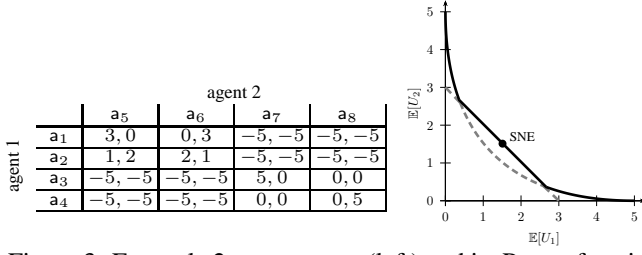| agent 1	a5	a6	a7	a8
a1	3, 0	0, 3	−5, −5	−5, −5
a2	1, 2	2, 1	−5, −5	−5, −5
a3	−5, −5	−5, −5	5, 0	0, 0
a4	−5, −5	−5, −5	0, 0	0, 5

Figure 3: Example 2–agent game (left) and its Pareto frontier (right).

Next we study the case of arbitrary–sized (potentially degenerate) games admitting SNEs with full supports.

Theorem 3.4 *Consider a 2–agent game with m_1 and m_2. If there is a mixed–strategy SNE with $|S_1| = m_1$ and $|S_2| = m_2$, then $P_{mix} = P_{cor}$.*

Proof sketch. The proof is similar to the proof of Theorem 3.1. Define $R_{i,j} = (U_1(i,j), U_2(i,j))$. By NE constraints and KKT conditions, we have that

- for all j, $\sum_{a_i} x_1(a_i) \cdot R_{i,j} = const$ (i.e., the convex combinations of the elements of each column of the bimatrix must have the same value), and
- for all i, $\sum_{a_j} x_2(a_j) \cdot R_{i,j} = const$ (i.e., the convex combinations of the elements of each column of the bimatrix must have the same value),

and we have that

- each convex combination is strictly inside the polygon whose vertices are points $R_{i,j}$ (because the combination is not degenerate), and
- the combination must be on a Pareto efficient edge (otherwise it would be Pareto dominated),

and therefore all the vertices must be aligned. In this way, the combination of all the points $R_{i,j}$ for every mixed strategy leads to a point that lies on the line connecting all the $R_{i,j}$. Thus we have $P_{mix} = P_{cor}$. □

The extension of Corollary 3.2 to the case with support size larger than two is very involved and we omit it here due to limited space. Instead, Corollary 3.3 easily extends to the case in which the support per agent is larger than two (we omit the proof because it is the same as that of Corollary 3.3).

Corollary 3.5 *Consider a 2–agent game with m_1 and m_2. If there is a mixed–strategy SNE with $|S_1| = \overline{m}_1$ and $|S_2| = \overline{m}_2$, then $P_{mix}(S_1, S_2) = P_{cor}(S_1, S_2)$.*

We will now discuss how the above results extend to more than two agents. For example, in the 3–agent setting, the vector of payoffs for each action profile is $R_{i,j,k} = (U_1(i,j,k), U_2(i,j,k), U_3(i,j,k))$. The crucial result is that necessary conditions, generated for only the actions in the supports, for mixed–strategy SNEs forced by NE constraints with KKT conditions for all the coalitions (i.e., $\{1,2\}$, $\{1,3\}$, $\{2,3\}$, $\{1,2,3\}$) require that all the $R_{i,j,k}$ lie on a plane (with n–agent games, all the payoff vectors restricted on S must lie on an $(n-1)$–dimensional hyperplane). Thus, we have:

Theorem 3.6 *Consider an n–agent game. If there is a mixed–strategy SNE with support profile S then $P_{mix}(S) = P_{cor}(S)$.*

The above theorem provides only some necessary conditions, given that many other conditions, due to, e.g., dominance, are required to have mixed–strategy SNEs. Interestingly, we can show that there are vectors of payoffs that satisfy all these conditions. For example, the following 3–agent game has a mixed–strategy SNE in which all the agents randomize with uniform probability over all their actions:

agent 2

agent 1	a4	a5	a6
a1	1, 0, 0	0, 1, 0	0, 0, 1
a2	0, 1, 0	0, 0, 1	1, 0, 0
a3	0, 0, 1	1, 0, 0	0, 1, 0

Agent 3 plays action a7

agent 2

agent 1	a4	a5	a6
a1	0, 1, 0	0, 0, 1	1, 0, 0
a2	0, 0, 1	1, 0, 0	0, 1, 0
a3	1, 0, 0	0, 1, 0	0, 0, 1

Agent 3 plays action a8

agent 2

agent 1	a4	a5	a6
a1	0, 0, 1	1, 0, 0	0, 1, 0
a2	1, 0, 0	0, 1, 0	0, 0, 1
a3	0, 1, 0	0, 0, 1	1, 0, 0

Agent 3 plays action a9

4 Worst–case 2–agent instances for support–enumeration algorithms

We now leverage the results from the previous section to show that it is possible to generate games, with m actions per agent, that have only one SNE, and the support size can be set anywhere in the range $\{1, \ldots, \lceil \frac{m}{2} \rceil\}$. Thus, given m, we can generate $O(4^{\frac{m}{2}})$ different game instances, each with a different SNE. These game instances are the worst–case instances for support–enumeration algorithms, given that, for each possible enumeration, it is possible to generate an instance that requires the algorithm to scan an exponential number of supports.

We first introduce some results that we subsequently exploit to generate our hard instances.

Corollary 4.1 *Given an even number $\overline{m}$, consider the following two–agent game in which each agent has $m = 2\overline{m}$ actions.*

$$U_i = \begin{bmatrix} U_i^{1,1} & U_i^{1,2} \\ U_i^{2,1} & U_i^{2,2} \end{bmatrix}$$

where $U_i^{j,k}$ are matrices $\overline{m} \times \overline{m}$ defined as follows:

$$U_1^{1,1}(i,k) = \begin{cases} 1 & \text{if } i+j \text{ odd} \\ 0 & \text{otherwise} \end{cases}$$

$$U_1^{2,1}(i,k) = \begin{cases} -\overline{m} & \text{if } i = j \\ 1 & \text{otherwise} \end{cases}$$

$$U_1^{1,2}(i,k) = -\overline{m}$$

$$U_1^{2,2}(i,k) = 0$$

$$U_2^{1,1}(i,k) = \begin{cases} 1 & \text{if } i+j \text{ even} \\ 0 & \text{otherwise} \end{cases}$$

$$U_2^{2,1} = U_1^{1,2}$$
$$U_2^{1,2} = U_1^{2,1}$$
$$U_2^{2,2} = U_1^{2,2}$$

This game has an SNE with support sizes $|S_1| = |S_2| = \overline{m}$ and no other SNE.

Proof. Denote by A_i^1 the first $\overline{m}$ actions of agent i and by A_i^2 the second $\overline{m}$ actions of agent i. For clarity we split the proof into two parts, discussed in the following paragraphs, respectively. An example game is in Fig. 4.
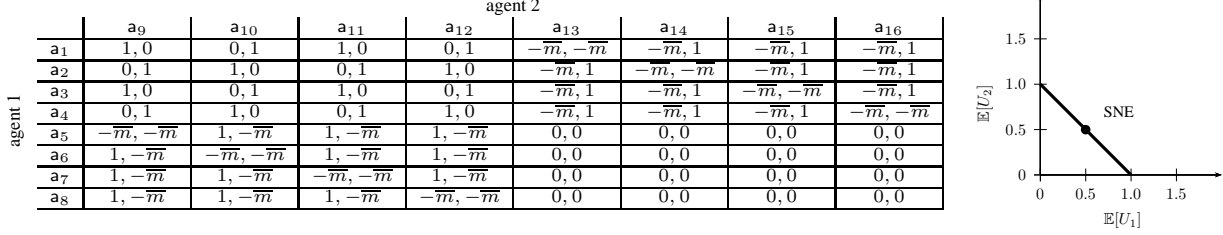
Figure 4: Example game that has an SNE with $|S_1| = |S_2| = 4 = \overline{m}$ (left), and its Pareto frontier (right).

There is an SNE with $|S_1| = |S_2| = \overline{m}$. The strategy profile in which each agent i plays all actions in A_i^1 with uniform probability $\frac{1}{\overline{m}}$ is an SNE. All the actions in the supports, i.e., A_i^1, provide utility $\frac{1}{2}$, while all the actions off the supports, i.e., A_i^2, provide utility $\frac{1 \cdot (\overline{m}-1) - \overline{m}}{\overline{m}} = -\frac{1}{\overline{m}}$. Therefore, it is an NE. Furthermore, the outcomes with utilities $(1, 0)$ and $(0, 1)$ are Pareto efficient, and the others are (weakly) dominated. Thus the Pareto frontier is a line that connects $(1, 0)$ to $(0, 1)$; therefore the NE is Pareto efficient. So, it is an SNE.

There is no other SNE. We focus on the strategy of agent 2 (the same reasoning can be applied for agent 1). Suppose agent 2 adopts a different strategy than the above uniform strategy over A_2^1. If agent 2 randomizes over a strict subset of A_2^1 composed of only odd actions or only even actions, then agent 1 has two best responses, one in A_1^1 and one in A_1^2 (each provides agent 1 utility 1). However, there is no NE of this form, given that, if agent 1 randomizes over such best responses, agent 2's best response would be to play some action in A_2^2 with a positive utility in place of a utility of $-\overline{m}$ given by the actions in A_2^1. If agent 2 randomizes over a strict subset of A_2^1 with even and odd actions, then agent 1 has one or more best responses in A_1^2 that provide utility 1 to agent 1. Also in this case, if agent 1 randomizes over such best responses, agent 2 would play some actions in A_2^2 gaining 0 in place of $-\overline{m}$ given by actions in A_2^1. If agent 2 randomizes over less than $\overline{m}$ actions in A_2^1 and A_2^2, then agent 1's best response is to play some action in A_1^2. Also in this case agent 2's best response would be to play some action in A_2^2. Finally, when both agents play only actions in A_i^2 or play more than $\overline{m}$ actions, we can have NEs, but these NEs are Pareto dominated (because they are randomizations over Pareto–dominated outcomes). $\qquad\square$

Games with $\overline{m}$ odd are a simple variation w.r.t. the even case (the proof is omitted being very similar):

Corollary 4.2 *Given an odd number $\overline{m}$, consider the following two–agent game in which each agent has $m = 2\overline{m}$ actions.*

$$U_i = \begin{bmatrix} U_i^{1,1} & U_i^{1,2} \\ U_i^{2,1} & U_i^{2,2} \end{bmatrix}$$

where $U_i^{j,k}$ are matrices $\overline{m} \times \overline{m}$ defined as follows:

- *$U_i^{1,2}$, $U_i^{2,1}$, $U_i^{2,2}$ are generated as in the case of $\overline{m}$ even,*

- *$U_i^{1,1}$ are composed of a submatrix $(\overline{m} - 1) \times (\overline{m} - 1)$ generated as in the case of $\overline{m}$ even and all the other entries are $\frac{1}{2}$.*

This game has an SNE with support sizes $|S_1| = |S_2| = \overline{m}$ and no other SNE.

We are now ready to define our hard–to–solve instances.

Definition 4.3 (Hard–to–solve instances) *Given m and $\overline{m}$, if $\overline{m} \in \{2, \ldots, \lfloor \frac{m}{2} \rfloor\}$, a hard–to–solve game instance is composed as follows:*

- *a sub–bimatrix of size $2\overline{m} \times 2\overline{m}$ built as described in Corollaries 4.1 and 4.2,*

- *all the other entries are drawn from $\{-\overline{m} \ldots, 0\}$ with uniform probability.*

If $\overline{m} = 1$, all the entries are drawn from $\{-\overline{m} \ldots, 0\}$ with uniform probability except for a pure action profile in which both agents have a utility of 1.

Theorem 4.4 *Hard–to–solve instances admit only one SNE.*

Proof sketch. The SNE described in Corollaries 4.1 and 4.2 is still an SNE because all the additional entries are smaller than the expected utility of the SNE. Because these additional entries are Pareto dominated, no additional SNE exists. $\qquad\square$

We leave open the extension to n–agent games.

5 Smoothed complexity of SNE finding

Worst–case complexity, being too pessimistic, is often a bad indicator of the actual performance of an algorithm, and average–case complexity is difficult to determine. A newer metric of complexity, called smoothed complexity, has been gaining interest in recent years [Bläser and Manthey, 2012]. It studies how the introduction of small perturbations affects the worst–case complexity. There might be several models of perturbations. By far the most common perturbation models are the uniform one and the Gaussian one.

In the case of the SNE-finding problem, given a perturbation $\mathcal{D}_\sigma$ of magnitude σ, these are defined as follows.

- Uniform perturbation: for each agent i, every entry in U_i is subjected to an additive perturbation $[-\sigma, +\sigma]$ with uniform probability.

- Gaussian perturbation: for each agent i, every entry in U_i is subjected to an additive perturbation $[-z, +z]$ with probability $\frac{1}{\sigma\sqrt{2\pi}} e^{-|U_i(j,k)-z|^2/\sigma^2}$.

Denote by $\tilde{U}_i$ the perturbed utility matrix.

We will first present results for the 2–agent setting. In the end of this section we show the generalization to any number of agents.

The smoothed running time of an algorithm $\mathcal{A}$ given a perturbation $\mathcal{D}_\sigma$ is defined as

$$\mathsf{smoothed}\text{–}t_{\mathcal{A}} = \mathbb{E}_{\tilde{U}_1, \tilde{U}_2 \sim \mathcal{D}_\sigma}[t_{\mathcal{A}}(\tilde{U}_1, \tilde{U}_2) | U_1, U_2]$$

where $t_\mathcal{A}(\tilde{U}_1, \tilde{U}_2)$ is the running time for the games instance $(\tilde{U}_1, \tilde{U}_2)$. An algorithm has smoothed polynomial time complexity if for all $0 < \sigma < 1$ there are positive constants c, k_1, k_2 such that:

$$\text{smoothed--}t_\mathcal{A} = O(c \cdot m^{k_1} \cdot \sigma^{-k_2})$$

where m is the size of the game in terms of actions per agent. Basically, a problem is in smoothed–$\mathcal{P}$ if it admits a smoothed polynomial time algorithm.

Theorem 5.1 *Finding an SNE is in smoothed–$\mathcal{P}$.*

Proof. We provide Algorithm 1. It has three main parts.

In the first part (Steps 1–3), the algorithm searches for a pure–strategy SNE by enumerating all the pure–strategy profiles and verifying whether each strategy profile is an SNE. The verification is accomplished by checking whether or not NE constraints are satisfied (this can be done in polynomial time in m) and by checking whether or not the strategy profile is on the Pareto frontier (this can be done in polynomial time as shown in [Gatti *et al.*, 2013]). If an SNE is found, the algorithm returns it. The maximum number of iterations in the first part of the algorithm is m^2.

In the second part (Steps 4–7), the algorithm verifies whether there there are strategy profile of support sizes $|S_1| + |S_2| = 3$ such that $P_{mix}(S_1, S_2) = P_{cor}(S_1, S_2)$. This can be accomplished in time $O(m^3)$ by checking whether there is a line connecting at least three entries of all the sub–bimatrices of size 2×2. In the affirmative case, the temporary variable temp is set true. Otherwise, temp is set false.

In the third part (Steps 8–13), if temp is false, the algorithm returns non–existence, given that, by Theorem 3.4, there is no mixed–strategy SNE. Otherwise, the algorithm enumerates all supports to find mixed–strategy NEs, and for each of them the algorithm verifies whether it is an SNE as done in Steps 1–3. In the latter case, the algorithm can take exponential time as discussed in the previous section.

Thus, the running time of Algorithm 1 is super–polynomial only if it needs to enumerate supports during Steps 8–13 (this can take exponential time). This happens only when $P_{mix}(S_1, S_2) = P_{cor}(S_1, S_2)$ for some S_1, S_2 with $|S_1| + |S_2| = 3$. Given that the perturbations $\mathcal{D}_\sigma$ over all the entries of the utility matrices are independent and identically distributed, the probability that the perturbed entries are aligned as required by Corollary 3.5 to have SNEs with $|S_1| = |S_2| > 1$ is zero. Therefore, the smoothed running time of Algorithm 1 is polynomial in m and independent of σ (for both uniform and Gaussian perturbations). $\qquad\square$

The above result shows that, except for a space of the parameters with zero measure, games admit only pure–strategy SNEs and therefore that verifying the existence of an SNE and finding them is computationally easy. Interestingly, the instability is due to the combination of NE constraints and Pareto efficiency constraints. Indeed, both the problem of finding an NE and the problem of finding Pareto efficient strategies are not sensitive to perturbations. Thus, while an approximate NE can be found by perturbing a game and finding an NE of the perturbed game (exactly, an NE of an ϵ–perturbed game is an 2ϵ–NE of the original game), this is not the case with SNE. Indeed, if a game admits only mixed–strategy SNEs, once perturbed, it does not admit any SNE.

Algorithm 1 SNEfinding

1: **for all** pure–strategy support profiles **do**
2:      **if** the support profile has an SNE **then**
3:          **return** the SNE
4: temp $\leftarrow$ false
5: **for all** strategy profiles with $|S_1| + |S_2| = 3$ **do**
6:      **if** payoffs restricted to supports are aligned **then**
7:          temp $\leftarrow$ true
8: **if** not temp **then**
9:      **return** non–existence
10: **else**
11:      **for all** support profiles **do**
12:          **if** the support profile has an SNE **then**
13:              **return** the SNE

The above results extend to the setting with any (constant) number of agents. As discussed in Section 3, in order to have mixed–strategy SNEs with n agents, all the payoffs vectors of the actions in the agents' supports must lie on the same $(n-1)$–hyperplane. Thus, if perturbed even with small perturbations, the probability that the payoff vectors satisfy such constraints is zero. Therefore, generic game instances (i.e., all instances except knife–edge cases) have only pure–strategy SNEs (if any).

6 Conclusions and future research

Strong Nash equilibrium (SNE) is the most natural solution concept for games where agents can form coalitions, i.e., coordinate their strategies. An SNE is a strategy profile where no coalition can deviate in a way that every one of the deviating agents (strictly) benefits. Given a finite game, a Nash equilibrium always exists, but an SNE might not exist. For finding an SNE, there has been a shortage of algorithms. Most algorithms only find pure–strategy SNEs in special game classes. A recent algorithm finds SNEs generally, but only in 2–agent games.

Our central result is that, in order for a game to have at least one mixed–strategy (i.e., non–pure–strategy) SNE, the agents' payoffs restricted to the agents' supports must, in the case of two agents, lie on the same line, and, in the case of n agents, lie on an $(n-1)$–dimensional hyperplane. Leveraging this result, we provided two contributions. First, we developed worst–case game instances for support–enumeration algorithms. These instances have only one SNE and the support size can be chosen to be of any size—in particular, arbitrarily large. In this way, for each possible enumeration, it is possible to generate an instance that requires the algorithm to scan an exponential number of supports. Second, we proved that, unlike Nash equilibrium, finding an SNE is in smoothed polynomial time: generic game instances (i.e., all instances except knife–edge cases) have only pure–strategy SNEs.

In future research we plan to study the computational complexity of approximating SNE and to design algorithms to do so. We also plan to study computational issues related to *strong correlated equilibrium*. This concept should present different properties than SNE, e.g., the convexity of the Pareto frontier with this solution concept could make the computation of an equilibrium easier and could make equilibria not sensitive to small perturbations.

References

[Aumann, 1960] R. Aumann. Acceptable points in games of perfect information. *PAC J MATH*, 10:381–417, 1960.

[Bläser and Manthey, 2012] M. Bläser and B. Manthey. Smoothed complexity theory. *CoRR*, abs/1202.1936, 2012.

[Chen *et al.*, 2006] X. Chen, X. Deng, and S.H. Teng. Computing Nash equilibria: approximation and smoothed complexity. In *FOCS*, pages 603–612, 2006.

[Chen *et al.*, 2009] X. Chen, X. Deng, and S.-H. Teng. Settling the complexity of computing two–player Nash equilibria. *J ACM*, 56(3):14:1–14:57, 2009.

[Conitzer and Sandholm, 2008] V. Conitzer and T. Sandholm. New complexity results about Nash equilibria. *GAME ECON BEHAV*, 63(2):621–641, 2008.

[Daskalakis *et al.*, 2006] C. Daskalakis, P. Goldberg, and C. Papadimitriou. The complexity of computing a Nash equilibrium. In *STOC*, pages 71–78, 2006.

[Epstein *et al.*, 2007] A. Epstein, M. Feldman, and Y. Mansour. Strong equilibrium in cost sharing connection games. In *ACM EC*, pages 84–92, 2007.

[Gatti *et al.*, 2012] N. Gatti, G. Patrini, M. Rocco, and T. Sandholm. Combining local search techniques and path following for bimatrix games. In *UAI*, pages 286–295, 2012.

[Gatti *et al.*, 2013] N. Gatti, M. Rocco, and T. Sandholm. On the verification and computation of strong Nash equilibrium. In *AAMAS*, 2013.

[Gourvès and Monnot, 2009] L. Gourvès and J. Monnot. On strong equilibria in the max cut game. In *WINE*, pages 608–615, 2009.

[Hayrapetyan *et al.*, 2006] A. Hayrapetyan, E. Tardos, and T. Wexler. The effect of collusion in congestion games. In *STOC*, pages 89–98, 2006.

[Hoefer and Skopalik, 2010] M. Hoefer and A. Skopalik. On the complexity of Pareto–optimal Nash and strong equilibria. In *SAGT*, pages 312–322, 2010.

[Holzman and Law-Yone, 1997] R. Holzman and N. Law-Yone. Strong equilibrium in congestion games. *GAME ECON BEHAV*, 21:85–101, 1997.

[Lemke and Howson, 1964] C. E. Lemke and J. J. T. Howson. Equilibrium points of bimatrix games. *SIAM J APPL MATH*, 12(2):413–423, 1964.

[Miettinen, 1999] K. Miettinen. *Multiobjective Optimization*. Kluwer, 1999.

[Nessah and Tian, 2012] R. Nessah and G. Tian. On the existence of strong Nash equilibria. *IESEG School of Management, Working Paper*, 2012.

[Nudelman *et al.*, 2004] E. Nudelman, J. Wortman, K. Leyton-Brown, and Y. Shoham. Run the GAMUT: A comprehensive approach to evaluating game–theoretic algorithms. In *AAMAS*, pages 880–887, 2004.

[Porter *et al.*, 2009] R. Porter, E. Nudelman, and Y. Shoham. Simple search methods for finding a Nash equilibrium. *GAME ECON BEHAV*, 63:642–662, 2009.

[Rozenfeld and Tennenholtz, 2006] O. Rozenfeld and M. Tennenholtz. Strong and correlated strong equilibria in monotone congestion games. In *WINE*, pages 74–86, 2006.

[Savani and von Stengel, 2006] R. Savani and B. von Stengel. Hard–to–solve bimatrix games. *ECONOMETRICA*, 74(2):397–429, 2006.

[Shoham and Leyton-Brown, 2008] Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game Theoretic and Logical Foundations*. Cambridge University Press, 2008.